What is claimed is:

1. A system architecture for adapting at least one legacy system for functional interface with at least one component system comprising:

a data reconciliation bus for data redundancy between legacy systems in the event of more than one legacy system;

at least one component wrapper within the architecture for describing the at least one legacy system;

at least one component object within the architecture for describing the at least one component system; and

a connectivity bus within the architecture between the at least one component object and the at least one component wrapper, for extending legacy function to the at least one component system;

characterized in that a user operating a GUI has access to legacy services in an automated client/server exchange wherein heterogeneous data formats and platform differences of the separate systems are resolved in an object-oriented way that is transparent to the user.

2. The system architecture of claim 1 wherein one component is interfaced with more than one legacy system in the event of more than one system.

3. The system architecture of claim 1 wherein one legacy system is interfaced with more than one component system in the event of more than one component system.

4. The system architecture of claim 1 wherein the data reconciliation bus utilizes an in memory entity-relationship model of each legacy system of the architecture.

5. The system architecture of claim 1 wherein entity-relationship modeling is used to model legacy services.

6. The system architecture of claim 5 wherein a component wrapper is completely generated from an object model of legacy services.

7. The system architecture of claim 1 wherein heterogeneity of data between a legacy system and a component wrapper is resolved by a language adapter interface.

8. A method for adapting at least one legacy system for functional interface with at least one component system comprising steps of:

(a) identifying the functionality of the at least one legacy system in terms of external expectations of the system;

(b) specifying individual services and aggregating them into a modeled set of services;

(c) generating an object facade from the service model that describes the aggregated legacy services;

(d) providing a component object that describes the functionality of the component system; and

(e) defining a mapping between the object facade and the component object.

9. The method of claim 8 wherein in step (b) services are expressed as n-tuples.

10. The method of claim 8 wherein in step (b) the service sets are modeled using object modeling.

11. The method of claim 8 wherein in step (c) the service model is an object model.

12. The method of claim 8 wherein in step (d) the component object is developed specifically for a corresponding object facade.

13. In a system architecture for integrating legacy systems and component systems, a data reconciliation framework for achieving data reconciliation between redundant data elements in the legacy systems comprising:

a memory component with a data model stored therein, the data model describing all legacy systems data and component systems data;

a first function for propagating data from a legacy system; and

a second function for propagating data to a legacy system or systems;

characterized in that the first function updates the data model and the second function takes the update from the data model as input and propagates it to the appropriate system or systems.

14. The data reconciliation framework of claim 13 wherein the data model stored in memory is a unified normalized layer.

15. The data reconciliation framework of claim 13 wherein the first and second functions are automated.

16. The data reconciliation framework of claim 13 wherein the first and second functions are user executed.

17. The data reconciliation framework of claim 13 wherein the functions propagate data in an object oriented environment.